

# Средства, применяемые при разработке программного обеспечения в ОС типа UNIX/Linux

---

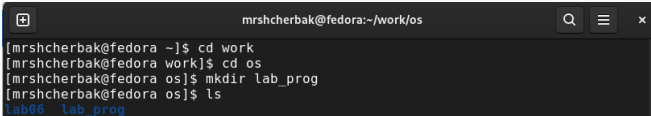
Щербак Маргарита Романовна

2022

RUDN

Приобрести простейшие навыки разработки, анализа, тестирования и отладки приложений в ОС типа UNIX/Linux на примере создания на языке программирования С калькулятора с простейшими функциями.

Ход работы: В домашнем каталоге создала подкаталог  
~/work/os/lab\_prog.(Рис. 1).

A terminal window with a dark background. The title bar shows the user 'mrshcherbak@fedora' and the current directory '~/work/os'. The terminal contains the following commands and output:

```
[mrshcherbak@fedora ~]$ cd work
[mrshcherbak@fedora work]$ cd os
[mrshcherbak@fedora os]$ mkdir lab_prog
[mrshcherbak@fedora os]$ ls
lab06  lab_prog
```

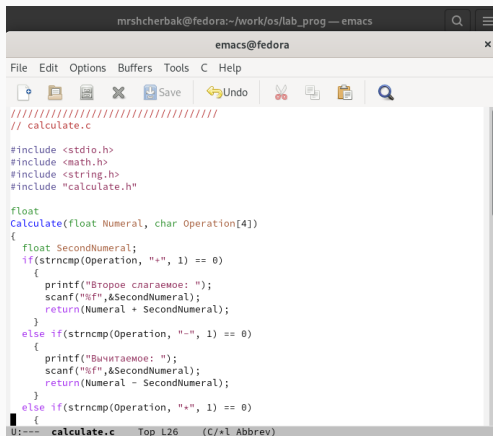
Figure 1: Создание каталога lab\_prog

Создала в нём файлы: calculate.h, calculate.c, main.c. (Рис. 2).

```
[mrshcherbak@fedora os]$ cd lab_prog  
[mrshcherbak@fedora lab_prog]$ touch calculate.h calculate.c main.c  
[mrshcherbak@fedora lab_prog]$ ls  
calculate.c calculate.h main.c  
[mrshcherbak@fedora lab_prog]$
```

Figure 2: Создание файлов

Открыла редактор emacs и записала реализацию функций калькулятора в файле calculate.c (Содержимое данного файла в полном объёме представлено в методичке к этой лабораторной работе): (Рис. 3).



```
mrshcherbak@fedora:~/work/os/lab_prog — emacs
emacs@fedora
File Edit Options Buffers Tools C Help
[Icons: New, Open, Save, Undo, Redo, Copy, Paste, Find]

////////////////////
// calculate.c

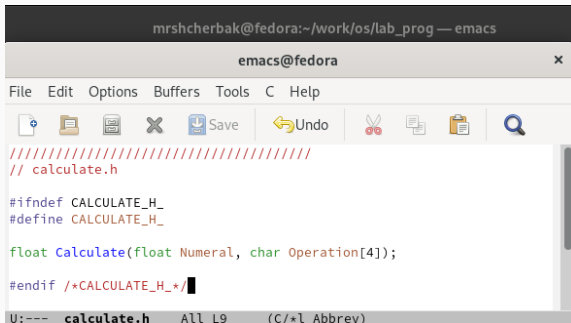
#include <stdio.h>
#include <math.h>
#include <string.h>
#include "calculate.h"

float
Calculate(float Numeral, char Operation[4])
{
    float SecondNumeral;
    if(strncmp(Operation, "+", 1) == 0)
    {
        printf("Второе слагаемое: ");
        scanf("%f", &SecondNumeral);
        return(Numeral + SecondNumeral);
    }
    else if(strncmp(Operation, "-", 1) == 0)
    {
        printf("Вычитаемое: ");
        scanf("%f", &SecondNumeral);
        return(Numeral - SecondNumeral);
    }
    else if(strncmp(Operation, "*", 1) == 0)
    {

```

Figure 3: Реализация функций калькулятора

В этом же редакторе открыла интерфейсный файл `calculate.h`, описывающий формат вызова функции калькулятора: (Рис. 4).



```
mrshcherbak@fedora:~/work/os/lab_prog — emacs
emacs@fedora
File Edit Options Buffers Tools C Help
[Icons: New, Open, Save, Close, Save, Undo, Cut, Copy, Paste, Search]
////////////////////////////////////
// calculate.h

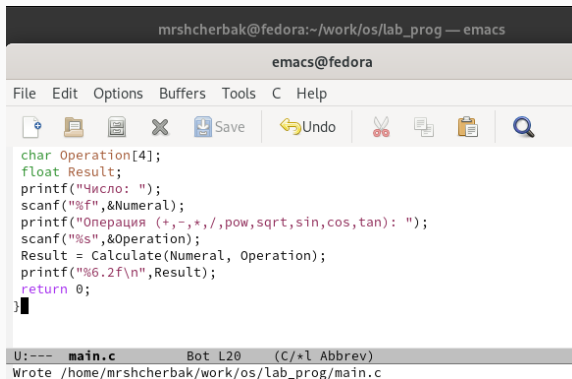
#ifndef CALCULATE_H_
#define CALCULATE_H_

float Calculate(float Numeral, char Operation[4]);

#endif /*CALCULATE_H_*/
U:--- calculate.h All L9 (C/*l Abbrev)
```

Figure 4: Содержимое файла `calculate.h`

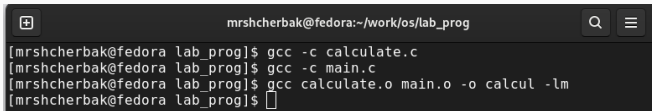
Аналогично открыла и написала основной файл main.c, реализующий интерфейс пользователя к калькулятору: (Рис. 5).



```
mrshcherbak@fedora: ~/work/os/lab_prog — emacs
emacs@fedora
File Edit Options Buffers Tools C Help
[Icons: Open, Save, Undo, Redo, Find]
char Operation[4];
float Result;
printf("Число: ");
scanf("%f",&Numeral);
printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
scanf("%s",&Operation);
Result = Calculate(Numeral, Operation);
printf("%6.2f\n",Result);
return 0;
}
U:--- main.c Bot L20 (C/*l Abbrev)
Wrote /home/mrshcherbak/work/os/lab_prog/main.c
```

Figure 5: Содержимое файла main.c

Выполнила компиляцию программы посредством gcc: (Рис.6).



```
mrshcherbak@fedora:~/work/os/lab_prog
[mrshcherbak@fedora lab_prog]$ gcc -c calculate.c
[mrshcherbak@fedora lab_prog]$ gcc -c main.c
[mrshcherbak@fedora lab_prog]$ gcc calculate.o main.o -o calcul -lm
[mrshcherbak@fedora lab_prog]$
```

Figure 6: Компиляция программы

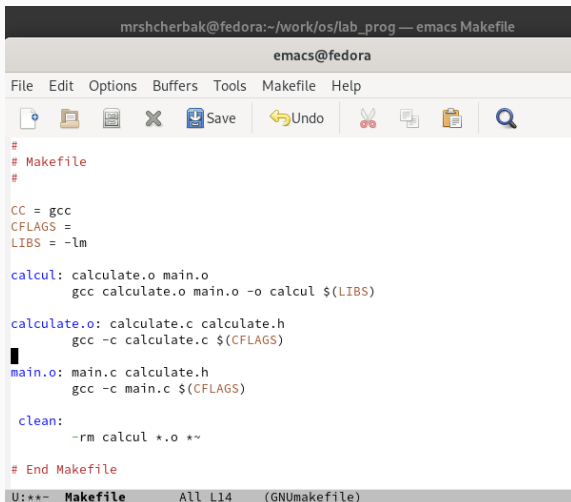


Создала Makefile со следующим содержанием: (Рис.7 - Рис.8).

```
[mrshcherbak@fedora lab_prog]$ touch Makefile
[mrshcherbak@fedora lab_prog]$ ls
calcul      calculate.c~ calculate.h~ main.c    main.o
calculate.c calculate.h  calculate.o main.c~   Makefile
[mrshcherbak@fedora lab_prog]$ emacs Makefile
█
```

Figure 7: Создание Makefile

# Makefile



```
mrshcherbak@fedora:~/work/os/lab_prog — emacs Makefile
emacs@fedora
File Edit Options Buffers Tools Makefile Help
[Icons: New, Open, Save, Undo, Redo, Find, etc.]

#
# Makefile
#

CC = gcc
CFLAGS =
LIBS = -lm

calcul: calculate.o main.o
    gcc calculate.o main.o -o calcul $(LIBS)

calculate.o: calculate.c calculate.h
    gcc -c calculate.c $(CFLAGS)

main.o: main.c calculate.h
    gcc -c main.c $(CFLAGS)

clean:
    -rm calcul *.o *~

# End Makefile

U:*** Makefile All L14 (GNUmakefile)
```

Figure 8: Содержимое Makefile

Перед использованием gdb исправила Makefile: в переменную CFLAGS добавила опцию -g, необходимую для компиляции объектных файлов и их использования в программе отладчика GDB. (Рис.9).

```
#  
# Makefile  
#  
CC = gcc  
CFLAGS = -g  
LIBS = -lm  
  
calcul: calculate.o main.o  
        gcc calculate.o main.o -o calcul $(LIBS)  
  
calculate.o: calculate.c calculate.h  
        gcc -c calculate.c $(CFLAGS)  
  
main.o: main.c calculate.h  
        gcc -c main.c $(CFLAGS)  
  
clean:  
        -rm calcul *.o *~  
  
# End Makefile
```

Figure 9: Исправленный Makefile

## С помощью gdb выполнила отладку программы calcul (Рис.10).

- запустила отладчик GDB, загрузив в него программу для отладки
- Для запуска программы внутри отладчика ввела команду run:

```
[mrshcherbak@fedora lab prog]$ gdb ./calcul
GNU gdb (GDB) Fedora 12.1-1.fc35
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software; you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./calcul...
(gdb) run
Starting program: /home/mrshcherbak/work/os/lab_prog/calcul

This GDB supports auto-downloading debuginfo from the following URLs:
https://debuginfod.fedoraproject.org/
Enable debuginfod for this session? (y or [n]) y
Debuginfod has been enabled.
To make this setting permanent, add 'set debuginfod enabled on' to .gdbinit.
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Число: 25
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): /
Делитель: 5
5.00
[Inferior 1 (process 13411) exited normally]
```

Figure 10: Ввели число 25, затем операцию деления и делитель 5. Получили ответ 5

- Для постраничного просмотра исходного кода использовала команду list.
- Для просмотра строк с 12 по 15 основного файла использовала list с параметрами: list 12,15.
- Для просмотра определённых строк не основного файла использовала list с параметрами: list calculate.c:20,29

```
mrshcherbak@fedora:~/work/os/lab_prog — gdb ./calcul
Делитель: 5
5.00
[Inferior 1 (process 13411) exited normally]
(gdb) list
1  //////////////////////////////////////////////////
2  // main.c
3
4  #include <stdio.h>
5  #include "calculate.h"
6
7  int
8  main (void){
9      float Numeral;
10     char Operation[4];
11
12     float Result;
13     printf("Число: ");
14     scanf("%f",&Numeral);
15     printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
16     scanf("%s",Operation);
17     Result = Calculate(Numeral, Operation);
18     printf("%.2f\n",Result);
19     return 0;
20 }
(gdb) list 12,15
12     printf("Число: ");
13     scanf("%f",&Numeral);
14     printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
15     scanf("%s",Operation);
(gdb) list calculate.c:20,29
20     {
21         printf("Вычитаемое: ");
22         scanf("%f",&SecondNumeral);
23         return(Numeral - SecondNumeral);
24     }
25     else if(strncmp(Operation, "**", 1) == 0)
26     {
27         printf("Множитель: ");
28         scanf("%f",&SecondNumeral);
29         return(Numeral * SecondNumeral);
(gdb) □
```

Figure 11: Команда list

## Выполнение нижеперечисленных действий: (Рис.12).

- установила точку останова в файле calculate.c на строке номер 21:  
list calculate.c:20,27  
break 21
- вывела информацию об имеющихся в проекте точках останова:  
info breakpoints
- запустила программу внутри отладчика и убедилась, что программа остановится в момент прохождения точки останова:  
run  
5  
“-” (просто минус)  
backtrace
- отладчик выдал следующую информацию:  
#0 Calculate (Numeral=5, Operation=0x7fffffffdec4 “-”) at calculate.c:21  
#1 0x0000000000400b2b in main () at main.c:16

```
mrshcherbak@fedora:~/work/os/lab_prog — gdb ./calcul
27     printf("Множитель: ");
28     scanf("%f",&SecondNumeral);
29     return(Numeral * SecondNumeral);
(gdb) list calculate.c:20,27
20     {
21     printf("Вычитаемое: ");
22     scanf("%f",&SecondNumeral);
23     return(Numeral - SecondNumeral);
24     }
25     else if(strncmp(Operation, "*", 1) == 0)
26     {
27     printf("Множитель: ");
(gdb) break 21
Breakpoint 1 at 0x40120f: file calculate.c, line 21.
(gdb) info breakpoints
Num  Type             Disp Enb Address            What
1     breakpoint       keep y   0x000000000040120f in Calculate at calculate.c:21
(gdb) run
Starting program: /home/mrshcherbak/work/os/lab_prog/calcul
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Число: 5
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): -

Breakpoint 1, Calculate (Numeral=5, Operation=0x7fffffffdec4 "-") at calculate.c:21
21     printf("Вычитаемое: ");
(gdb) backtrace
#0 Calculate (Numeral=5, Operation=0x7fffffffdec4 "-") at calculate.c:21
#1 0x00000000004014eb in main () at main.c:16
(gdb) □
```

Figure 12: Выполнение



## выполнение нижеперечисленных действий (Рис.13).

Посмотрела, чему равно на этом этапе значение переменной Numeral,  
введя: `print Numeral`

На экран должно быть выведено число 5.

Сравнила с результатом вывода на экран после использования  
команды: `display Numeral`

Убрала точки останова:

`info breakpoints`

`delete 1`

```
(gdb) print Numeral
$1 = 5
(gdb) display Numeral
1: Numeral = 5
(gdb) info breakpoints
Num      Type           Disp Enb Address            What
1        breakpoint     keep y   0x000000000040120f in calculate at calculate.c:21
          breakpoint already hit 1 time
(gdb) delete 1
```

Figure 13: Выполнение

## С помощью утилиты splint попробовала проанализировать коды файлов calculate.c и main.c. (Рис.14 - Рис.15).

```
mrshcherbak@fedora:~/work/os/lab_prog
[mrshcherbak@fedora lab_prog]$ splint calculate.c
Splint 3.1.2 --- 23 Jul 2021

calculate.h:7:37: Function parameter Operation declared as manifest array (size
        constant is meaningless)
    A formal parameter is declared as an array with size. The size of the array
    is ignored in this context, since the array formal parameter is treated as a
    pointer. (Use -fixedformalarray to inhibit warning)
calculate.c:10:31: Function parameter Operation declared as manifest array
        (size constant is meaningless)
calculate.c: (in function calculate)
calculate.c:16:7: Return value (type int) ignored: scanf("%f", &Sec...
    Result returned by function call is not used. If this is intended, can cast
    result to (void) to eliminate message. (Use -retvalint to inhibit warning)
calculate.c:22:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:28:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:34:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:35:10: Dangerous equality comparison involving float types:
        SecondNumeral == 0
    Two real (float, double, or long double) values are compared directly using
    == or != primitive. This may produce unexpected results since floating point
    representations are inexact. Instead, compare the difference to FLT_EPSILON
    or DBL_EPSILON. (Use -realcompare to inhibit warning)
calculate.c:38:13: Return value type double does not match declared type float:
        (HUGE_VAL)
    To allow all numeric types to match, use -relaxtypes.
calculate.c:46:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:47:13: Return value type double does not match declared type float:
        (pow(Numeral, SecondNumeral))
calculate.c:50:11: Return value type double does not match declared type float:
        (sqrt(Numeral))
calculate.c:52:11: Return value type double does not match declared type float:
        (sin(Numeral))
calculate.c:54:11: Return value type double does not match declared type float:
        (cos(Numeral))
calculate.c:56:11: Return value type double does not match declared type float:
        (tan(Numeral))
calculate.c:60:13: Return value type double does not match declared type float:
        (HUGE_VAL)

Finished checking --- 15 code warnings
```

Figure 14: Splint calculate.c

```
[mrshcherbak@fedora lab_prog]$ splint main.c
Splint 3.1.2 --- 23 Jul 2021

calculate.h:7:37: Function parameter Operation declared as manifest array (size
        constant is meaningless)
    A formal parameter is declared as an array with size. The size of the array
    is ignored in this context, since the array formal parameter is treated as a
    pointer. (Use -fixedformalarray to inhibit warning)
main.c: (in function main)
main.c:13:2: Return value (type int) ignored: scanf("%f", &Num...
    Result returned by function call is not used. If this is intended, can cast
    result to (void) to eliminate message. (Use -retvalint to inhibit warning)
main.c:15:13: Format argument 1 to scanf ("%s") expects char * gets char [4] *:
    &Operation
    Type of parameter is not consistent with corresponding code in format string.
    (Use -formattype to inhibit warning)
    main.c:15:10: Corresponding format code
main.c:15:2: Return value (type int) ignored: scanf("%s", &0pe...

Finished checking --- 4 code warnings
[mrshcherbak@fedora lab_prog]$
```

Figure 15: Splint main.c

Таким образом, в ходе ЛРН№13 я приобрела простейшие навыки разработки, анализа, тестирования и отладки приложений в ОС типа UNIX/Linux на примере создания на языке программирования С калькулятора с простейшими функциями.